RSAC 2025 Conference

Many Voices. One Community.

SESSION ID: CRYP-M05A

How (Not) to Hash into Class Groups of Imaginary Quadratic Fields?

István András Seres[1], Péter Burcsi[1], Péter Kutas[1,2]

1 Eötvös Loránd University, Hungary, Budapest
2 University of Birmingham, UK, Birmingham

Disclaimer

Presentations are intended for educational purposes only and do not replace independent professional judgment. Statements of fact and opinions expressed are those of the presenters individually and, unless expressly stated to the contrary, are not the opinion or position of RSA Conference LLC or any other co-sponsors. RSA Conference LLC does not endorse or approve, and assumes no responsibility for, the content, accuracy or completeness of the information presented.

Attendees should note that sessions may be audio- or video-recorded and may be published in various media, including print, audio and video formats without further notice. The presentation template and any media capture are subject to copyright protection.

© 2025 RSA Conference LLC or its affiliates. The RSAC and RSAC CONFERENCE logos and other trademarks are proprietary. All rights reserved.

Hashing to (class) groups?! Why care?

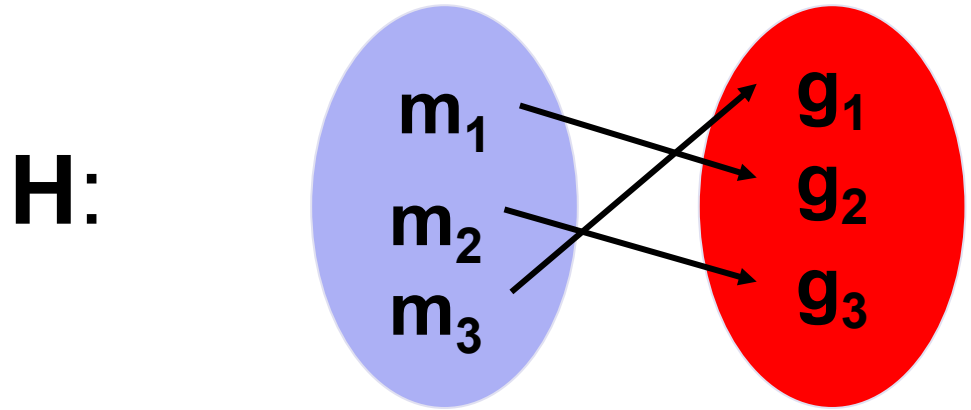
- **Many applications** need a secure hash function to class groups!

Hashing to (class) groups?! Why care?

- **Many applications** need a secure hash function to class groups!
  - Verifiable delay functions [BBBF18],
  - Zero-knowledge proofs [BFS20],
  - Timed commitments [TCLM21], and more!

Hashing to (class) groups?! Why care?

- **Many applications** need a secure hash function to class groups!
  - Verifiable delay functions [BBBF18],
  - Zero-knowledge proofs [BFS20],
  - Timed commitments [TCLM21], and more!
- But, it's often **tricky mapping some message** *m* to a group **G**.



Why are class groups great in crypto?

RSA groups

Class groups

Why are class groups great in crypto?

RSA groups

- For a group $\mathbf{G}=\mathbf{Z}^*_N$:
  - G's order is unknown ***only if certain conditions are met!***

Class groups

- For a class group G:
  - G's order is unknown ***for everybody!***

Why are class groups great in crypto?

RSA groups

- For a group $\mathbf{G}=\mathbf{Z}^*_N$:
  - **G**'s order is unknown ***only if certain conditions are met!***
    - Let **N**=*pq*, for large *p,q* primes
    - **G**'s order only unknown if factoring is hard AND *p,q* are kept secret!

Class groups

- For a class group **G**:
  - **G**'s order is unknown ***for everybody!***

Why are class groups great in crypto?

RSA groups

- For a group $\mathbf{G}=\mathbf{Z}^*_N$:
  - G's order is unknown ***only if certain conditions are met!***
    - Let $\mathbf{N}=pq$, for large p,q primes
    - G's order only unknown if factoring is hard AND p,q are kept secret!
- **Trusted setup!** (MPC needed!)

Class groups

- For a class group G:
  - G's order is unknown ***for everybody!***
- **Transparent setup!** (No MPC!)

Why are class groups great in crypto?

RSA groups

- For a group $\mathbf{G}=\mathbf{Z}^*_N$:
  - G's order is unknown ***only if certain conditions are met!***
    - Let $\mathbf{N}=pq$, for large p,q primes
    - G's order only unknown if factoring is hard AND p,q are kept secret!
- **Trusted setup!** (MPC needed!)
- Ring elements: residues mod **N**.

Class groups

- For a class group G:
  - G's order is unknown ***for everybody!***
- **Transparent setup!** (No MPC!)
- Binary quadratic forms (see later)

Why are class groups great in crypto?

RSA groups

- For a group $\mathbf{G}=\mathbf{Z}^*_N$:
  - G's order is unknown ***only if certain conditions are met!***
    - Let $\mathbf{N}=pq$, for large p,q primes
    - G's order only unknown if factoring is hard AND p,q are kept secret!
- **Trusted setup!** (MPC needed!)
- Ring elements: residues mod **N**.
- For 128-bit security $\log_2\mathbf{N}\approx 3072$.

Class groups

- For a class group G:
  - G's order is unknown ***for everybody!***
- **Transparent setup!** (No MPC!)
- Binary quadratic forms (see later)
- For 128-bit security $\log_2\mathbf{\Delta}\approx 6800$.

Why are class groups great in crypto?

RSA groups

- For a group $\mathbf{G}=\mathbf{Z}^*_N$:
  - G's order is unknown ***only if certain conditions are met!***
    - Let $\mathbf{N}=pq$, for large p,q primes
    - G's order only unknown if factoring is hard AND p,q are kept secret!
- **Trusted setup!** (MPC needed!)
- Ring elements: residues mod **N**.
- For 128-bit security $\log_2\mathbf{N}\approx 3072$.
- *Karatsuba* multiplication algorithm for efficient group operations.

Class groups

- For a class group G:
  - G's order is unknown ***for everybody!***
- **Transparent setup!** (No MPC!)
- Binary quadratic forms (see later)
- For 128-bit security $\log_2\mathbf{\Delta}\approx 6800$.
- *Shanks'* NUCOMP and NUDPL algorithms are used for efficient class group operations.

Our contributions

https://github.com/seresistvanandras/hashingToClassGroups

Our contributions

- We identify two families of **broken** hash functions to class groups.
  - SageMath's hash function to class groups is insecure.
  - If deployed in certain applications, then the application would break.

https://github.com/seresistvanandras/hashingToClassGroups

Our contributions

- We identify two families of **broken** hash functions to class groups.
  - SageMath's hash function to class groups is insecure.
  - If deployed in certain applications, then the application would break.
- We identify two families of **secure** hash functions to class groups.
  - An algebraic hash function.
  - A surjective extension to Wesolowski's hash function to class groups.

https://github.com/seresistvanandras/hashingToClassGroups

Our contributions

- We identify two families of **broken** hash functions to class groups.
  - SageMath's hash function to class groups is insecure.
  - If deployed in certain applications, then the application would break.
- We identify two families of **secure** hash functions to class groups.
  - An algebraic hash function.
  - A surjective extension to Wesolowski's hash function to class groups.
- Open-source **implementation** and performance evaluation.

https://github.com/seresistvanandras/hashingToClassGroups

Agenda

- **Preliminaries** on class groups of imaginary quadratic fields
  - What is a binary quadratic form (a,b)?
  - What is the distribution of the coefficients *a* and *b*?
- **How Not to Hash** into Class Groups?
  - SageMath's hash function (inappropriate for crypto applications).
  - An attack for a VDF application using this broken hash function.
- **How to Hash** into Class Groups?
  - A simple algebraic hash function.
  - Our extended hash function of Wesolowski's hash function.
- **Future directions**, Open Questions

RSAC 2025 Conference

What is a Class Group of Imaginary Quadratic Fields?

What is a class group? I.

- A binary quadratic form is a function $f(X,Y)=aX^2+bXY+cY^2$.
- The *discriminant* **Δ** of a binary quadratic form *f* is $\Delta(f):=b^2-4ac$.
- Let us fix a prime **Δ**.
  - For instance, **Δ=-23**.
- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. $\Delta=b^2-4ac$.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}. Easy to see: $a,b\leq\Delta^{1/2}/3$.
- Typically we **represent group elements** as *(a,b)*. No need for *c*.
  - One can further compress group elements, see [DGS21].
- There is a group structure on Cl(**Δ**) due to Gauss (1801).
- Dirichlet (1839): $|Cl(\Delta)|\approx O(|\Delta|^{1/2})$.
- **No efficient algorithm** to compute the order of Cl(**Δ**).

What is a class group? II.

What is a class group? II.

- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. **Δ**=b^2-*4ac*.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}.

What is a class group? II.

- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. **Δ**=b^2-*4ac*.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}.
- Wait? Are not there more elements in Cl(**-23**)?
  - Let *g*=(1,3,8), then **Δ**(*g*)=*9-4*8=-23*. Yes, but…

What is a class group? II.

- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. **Δ**=b^2-4*ac*.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}.
- Wait? Are not there more elements in Cl(**-23**)?
  - Let *g*=(1,3,8), then **Δ**(*g*)=*9-4*8=-23*. Yes, but…
- *f*=(1,1,6) and *g*=(1,3,8) are equivalent, since $f(X,Y)=X^2+XY+6Y^2$ transformed by $f(X+Y,Y)=(X+Y)^2+(X+Y)Y+6Y^2=g(X,Y)=(1,3,8)$.

What is a class group? II.

- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. **Δ**=b^2-4*ac*.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}.
- Wait? Are not there more elements in Cl(**-23**)?
  - Let *g*=(1,3,8), then **Δ**(*g*)=*9-4*8=-23*. Yes, but…
- *f*=(1,1,6) and *g*=(1,3,8) are equivalent, since $f(X,Y)=X^2+XY+6Y^2$ transformed by $f(X+Y,Y)=(X+Y)^2+(X+Y)Y+6Y^2=g(X,Y)=(1,3,8)$.
- *f* is called a reduced binary form iff.-*a*<*b*≤*a* and *a*<*c*.

What is a class group? II.

- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. **Δ**=b^2-4*ac*.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}.
- Wait? Are not there more elements in Cl(**-23**)?
  - Let *g*=(1,3,8), then **Δ**(*g*)=*9-4*8=-23*. Yes, but…
- *f*=(1,1,6) and *g*=(1,3,8) are equivalent, since $f(X,Y)=X^2+XY+6Y^2$ transformed by $f(X+Y,Y)=(X+Y)^2+(X+Y)Y+6Y^2=g(X,Y)=(1,3,8)$.
- *f* is called a reduced binary form iff.-*a*<*b*≤*a* and *a*<*c*.
- Binary quadratic forms by **linear transformations** are equivalent.

What is a class group? II.

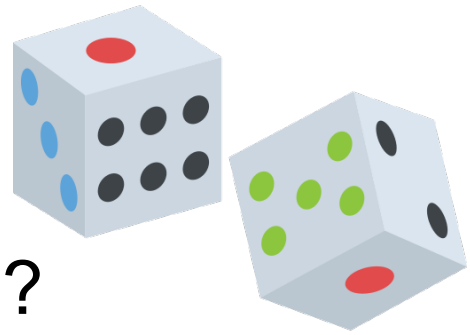
- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. **Δ**=b^2-4*ac*.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}.
- Wait? Are not there more elements in Cl(**-23**)?
  - Let *g*=(1,3,8), then **Δ**(*g*)=*9-4*8=-23*. Yes, but…
- *f*=(1,1,6) and *g*=(1,3,8) are equivalent, since $f(X,Y)=X^2+XY+6Y^2$ transformed by $f(X+Y,Y)=(X+Y)^2+(X+Y)Y+6Y^2=g(X,Y)=(1,3,8)$.
- *f* is called a reduced binary form iff.$-a<b\leq a$ and $a<c$.
- Binary quadratic forms by **linear transformations** are equivalent.
- More precisely, Cl(**Δ**) is the **set of reduced** binary quadratic forms.

What is a class group? II.

- Cl(**Δ**) is the set of *integer* triplets *(a,b,c)* s.t. **Δ**=b^2-4*ac*.
  - Cl(**-23**)={(1,1,6),(2,1,3),(2,-1,3)}.
- Wait? Are not there more elements in Cl(**-23**)?
  - Let *g*=(1,3,8), then **Δ**(*g*)=*9-4*8=-23*. Yes, but…
- *f*=(1,1,6) and *g*=(1,3,8) are equivalent, since $f(X,Y)=X^2+XY+6Y^2$ transformed by $f(X+Y,Y)=(X+Y)^2+(X+Y)Y+6Y^2=g(X,Y)=(1,3,8)$.
- *f* is called a reduced binary form iff.$-a<b\leq a$ and $a<c$.
- Binary quadratic forms by **linear transformations** are equivalent.
- More precisely, Cl(**Δ**) is the **set of reduced** binary quadratic forms.
- Analogy: equivalent residue classes mod **N**. 3≡11 mod **8**.

Desiderata for hashing to class groups

- Cl(**Δ**): the set of *reduced* binary quadratic forms *(a,b,c)* s.t. **Δ**=b^2-4ac.
  - Cl(**-47**)={(1,1,12),(2,1,6),(2,-1,6),(3,-1,4),(3,1,4)}.
- (Recall a,b≤$\mathbf{\Delta}^{1/2}$/3)
- What do we want from a hash-to-class group function **H**?
  - *Deterministic*: **H** is a deterministic function.
  - *Efficient*: computing **H** can be done fast.
  - *Collision resistance*: finding m_1,m_2 such that **H**(m_1)=**H**(m_2) is hard.
  - *One-way function*: given **H**(m_1), it is hard to compute m_1.
  - *Uniformly distributed*: the codomain of **H** is uniform.

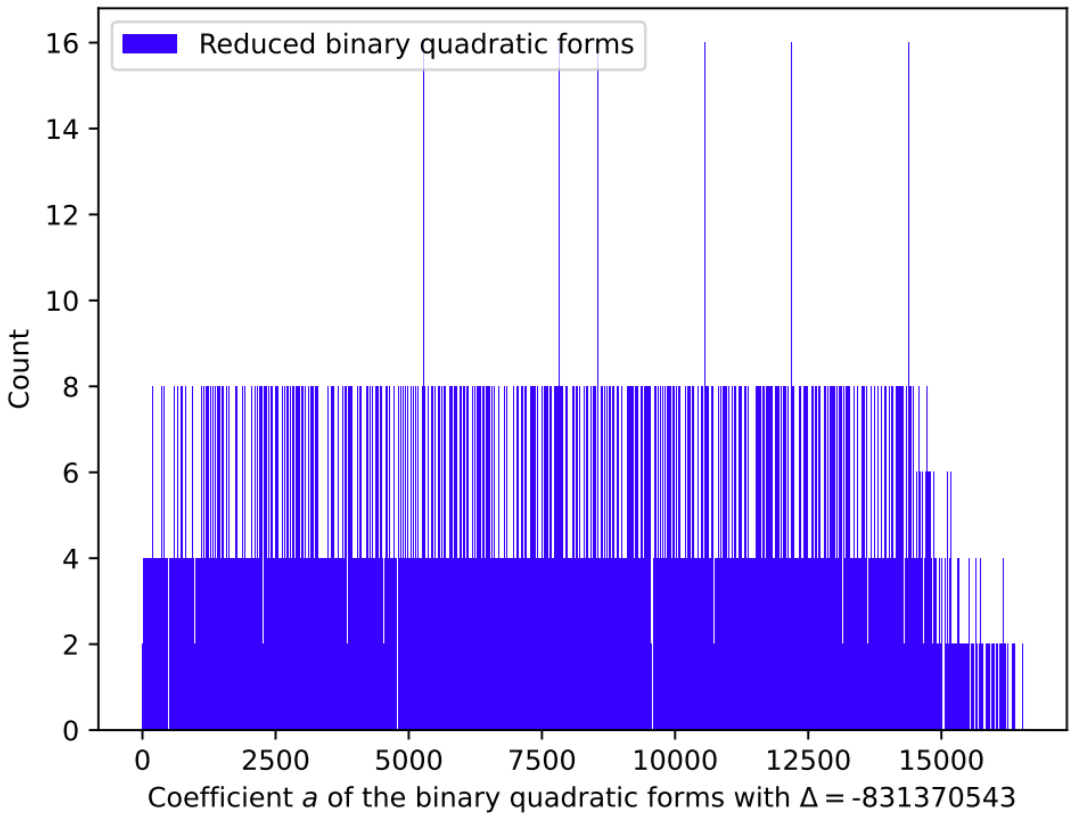


RSAC 2025 Conference

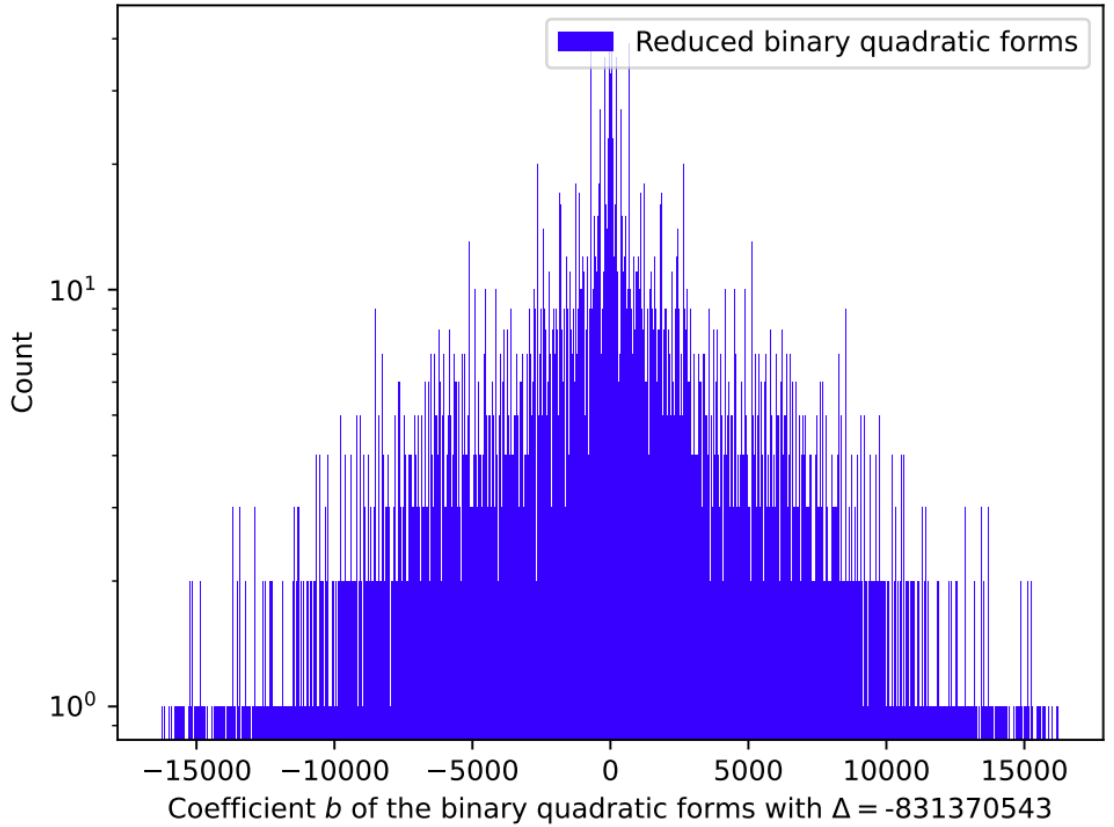
How Not to Hash into Class Groups of Imaginary Quadratic Fields?

Distribution of (a,b) in Cl(-831370543)

Distribution of coefficient “a”



Distribution of coefficient “b”



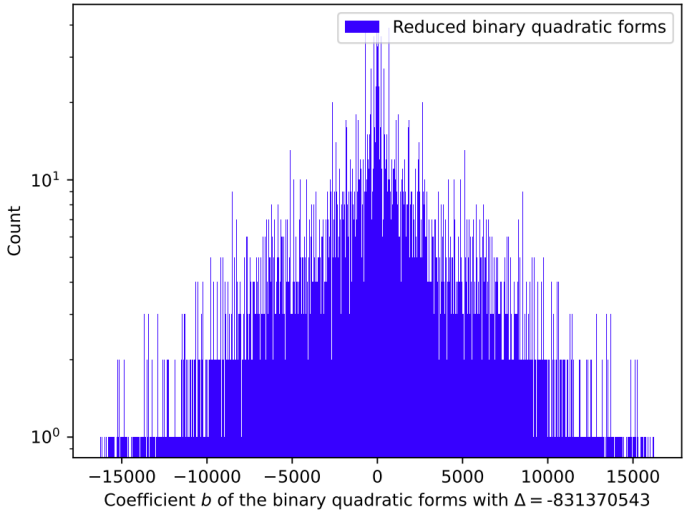
A conjecture for number theorists

Conjecture (The distribution of "b" in Cl(Δ))

- For a fixed **Δ**, there exist constants c_1, c_2 such that for any large enough *B*, the average number of forms in Cl(**Δ**) having a coefficient *b* in [B,2B) follows asymptotically

$$c_1 \log(|\mathbf{\Delta}|) - c_2 \log(b) + O(1),$$

whenever $\mathbf{\Delta} \to -\infty$.



How (Not) to Hash into Class Groups? I.

How (Not) to Hash into Class Groups? I.

1. Sampling a random coefficient “*b*”?
   a. Non-uniform distribution.

How (Not) to Hash into Class Groups? I.

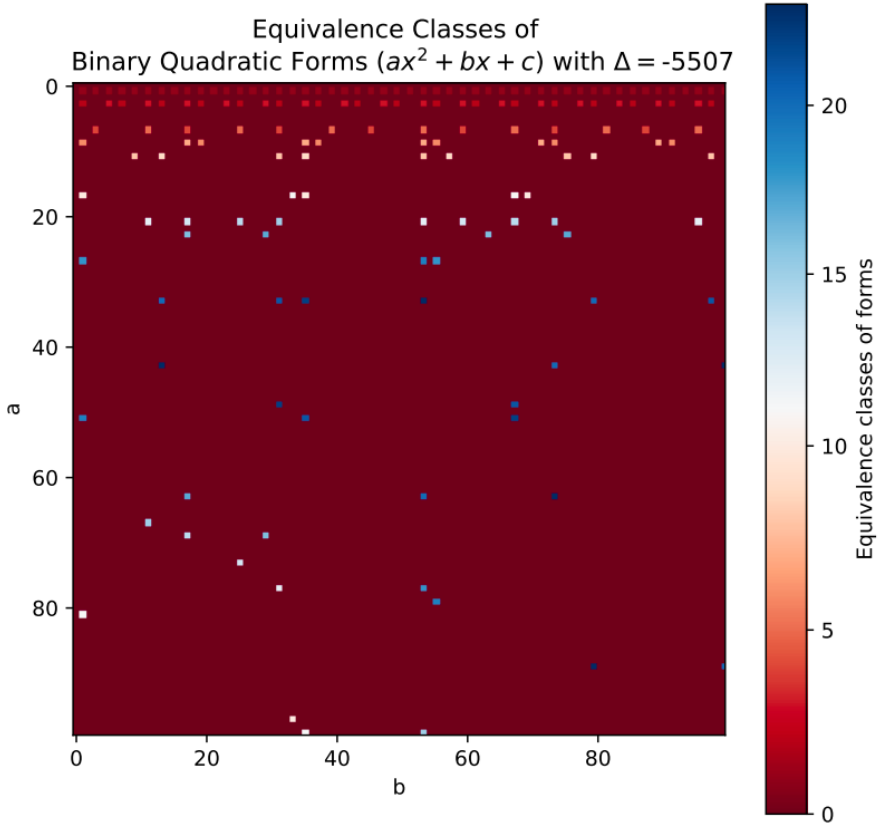
1. Sampling a random coefficient “*b*”?
   a. Non-uniform distribution.
2. Sampling a random coefficient “*a*”?
   a. Non-uniform distribution

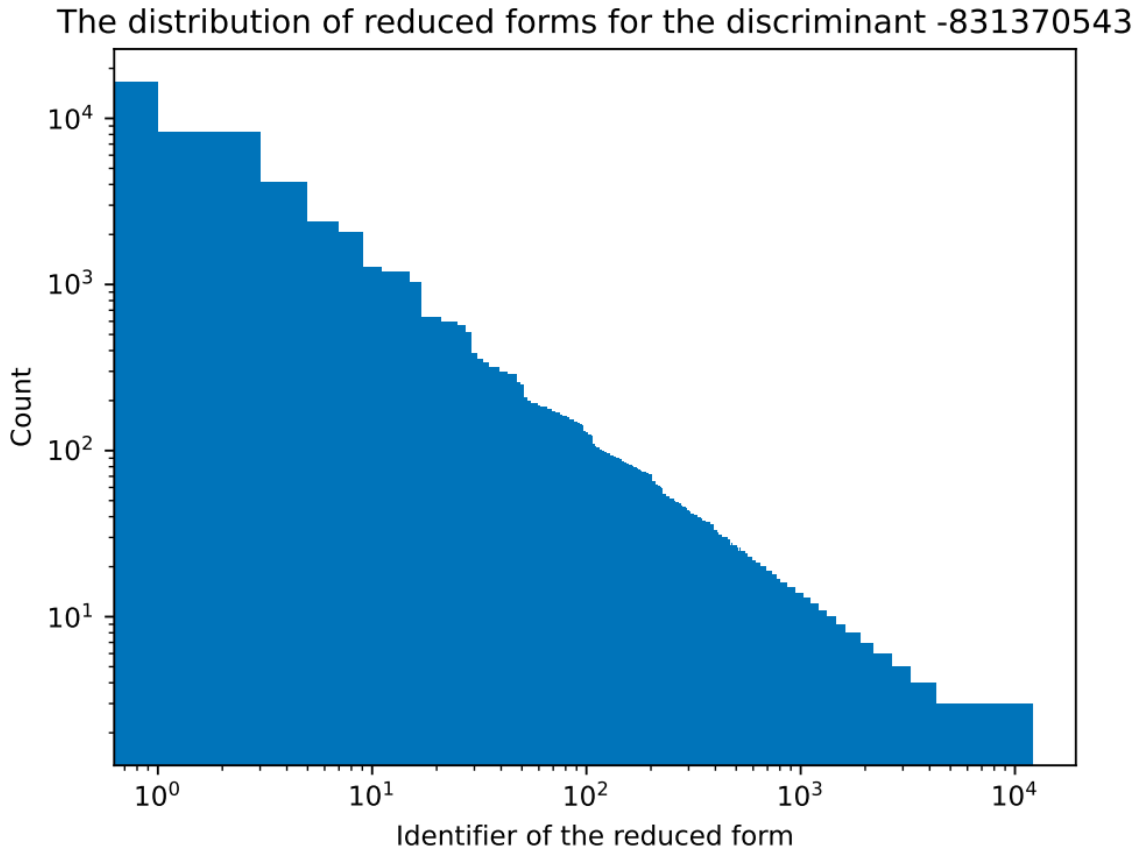
How (Not) to Hash into Class Groups? I.

1. Sampling a random coefficient “*b*”?
   a. Non-uniform distribution.
2. Sampling a random coefficient “*a*”?
   a. Non-uniform distribution
3. For an input message *m*, let **H**(m):=f^m.
   a. In many applications, unknown DLOG is necessary.

How (Not) to Hash into Class Groups? II.

- Random form and reduce? *(Implemented by SageMath)*





RSAC 2025 Conference

How to Hash into Class Groups of Imaginary Quadratic Fields?

How to Hash into Class Groups? I.

How to hash into class groups? Construction #1.

Public parameters: $\mathbb{G} \xleftarrow{R} GGen(\lambda), \mathsf{H}_0 : \{0,1\}^* \to \{0,1\}^d$ a secure hash function. A precomputation outputs $\{f_i\}_{i=1}^d \in \mathsf{Cl}(\Delta)^d$ forms with small prime norms, i.e., $\forall i \in [1,d] : f_i = (p_i, b_i) \in \mathsf{Cl}(\Delta)$ for some "small" prime $p_i \in [2, \mathcal{O}(\log|\Delta|)]$.
Inputs: $x \in \{0,1\}^*$.

1. Let $b := \mathsf{H}_0(x)$ and b_i be the ith bit of b.
2. Return: $H(x) := \prod_{i=1}^{d} f_i^{b_i} (\in \mathsf{Cl}(\Delta))$.

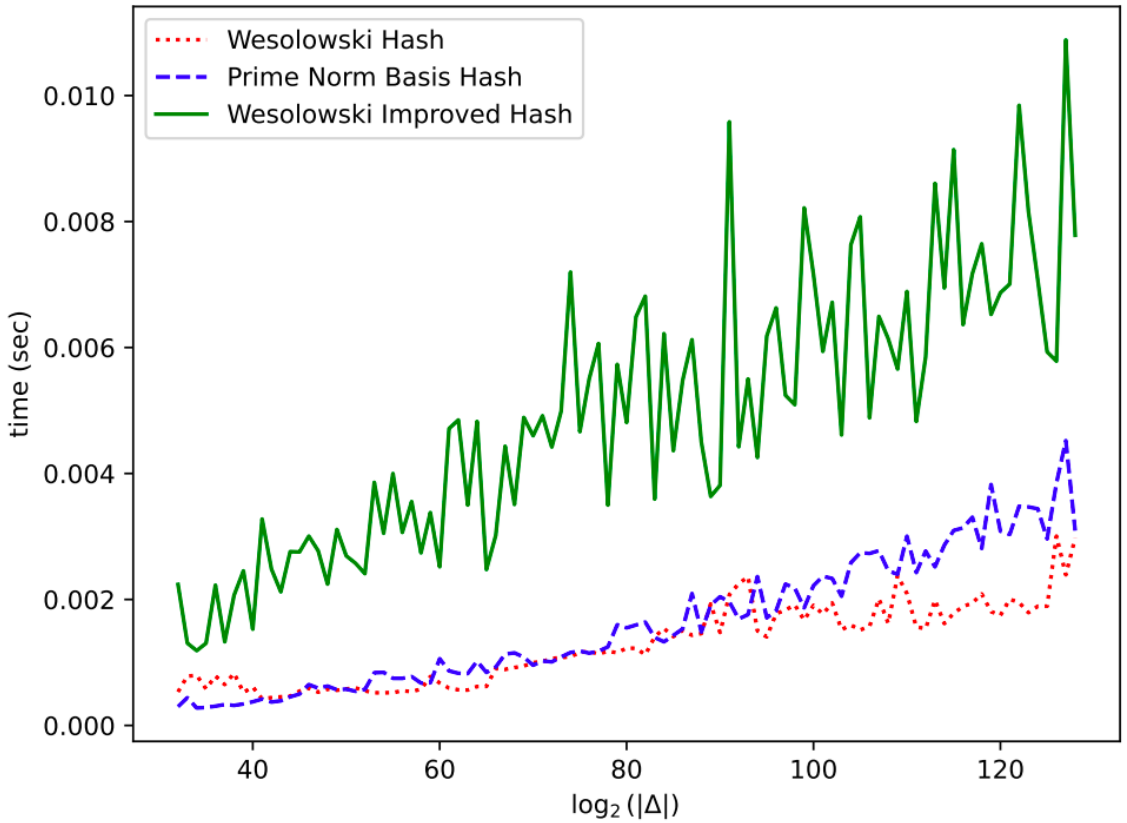
How to Hash into Class Groups? II.

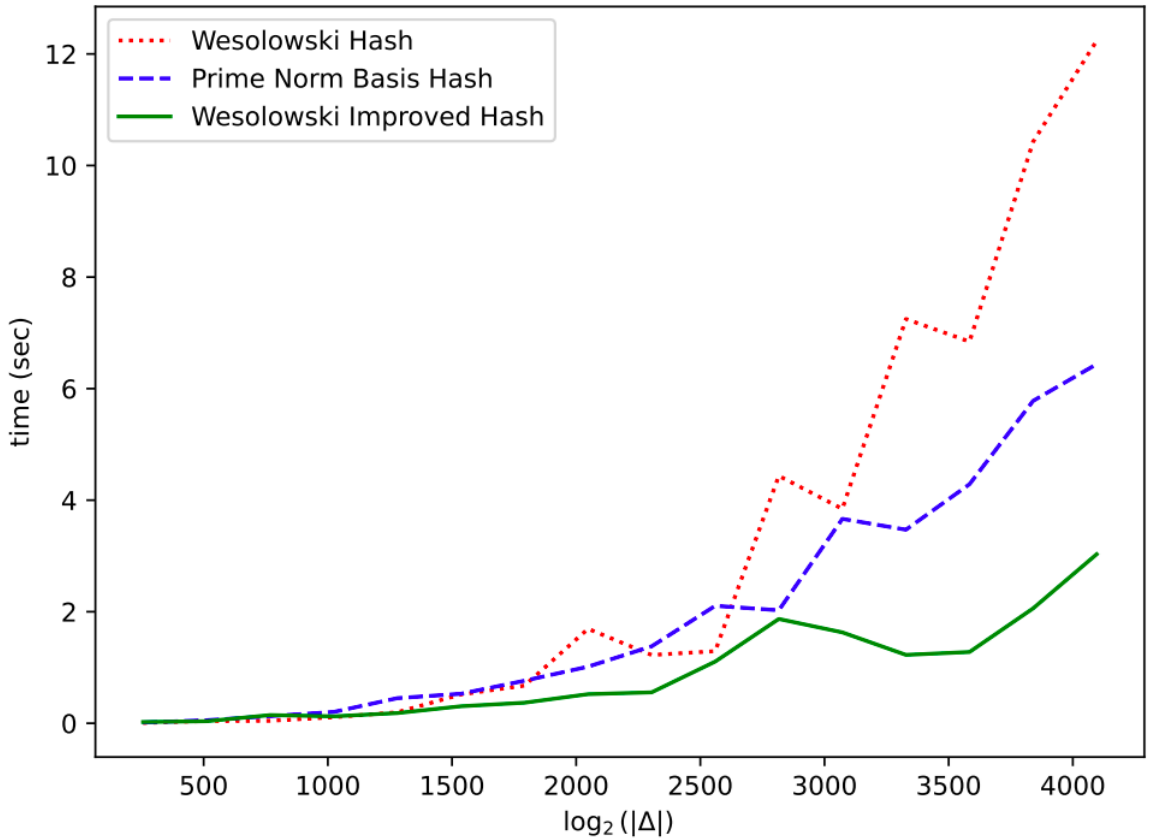
How to hash into class groups? Construction #2.

Public parameters: $\mathbb{G} \xleftarrow{R} GGen(\lambda), \mathsf{H}_{\mathsf{Primes}} : \{0,1\}^* \to \mathsf{Primes}(\lambda)$ a cryptographically secure hash function. Let Δ be the discriminant corresponding to the generated group $\mathbb{G}$.
Inputs: $x \in \{0,1\}^*$.

1. $i = 0$.
2. Compute $p := \mathsf{H}_{\mathsf{Primes}}(x||i)$. // For the sake of uniform distribution p must be large, i.e., $p \xleftarrow{\$} [0, \sqrt{|\Delta|/4}]$. We also need to ensure the reducedness of the form.
3. If $\left(\frac{\Delta}{p}\right) = 1$ then let $b := (\sqrt{\Delta} \bmod p)$.
   - else $i = i + 1$, and *goto* 2. //There is no prime form $(p, \cdot) \in \mathsf{Cl}(\Delta)$, cf. Theorem 1.
4. Let $s := (p \mod 3)$. // If $\left(\frac{\Delta}{p}\right) = 1$, then there are two modular square roots: b and $-b$. We need to choose one of them deterministically.
   - If $s == 0 \vee s == 1$: Return $H(x) := (p, b)$.
   - elif $s == 2$: Return $H(x) := (p, -b)$.

Performance Evaluation





RSAC 2025 Conference

Open Problems and Future Research Directions

Future Work and Research Directions

- More efficient hash-to-class group functions?
  - More efficient and constant time *hash-to-prime functions*?
  - Efficiency in an arithmetic circuit computation model, e.g., ZKPs, MPC.
- **Constant time** hash-to-class group functions?
- **More applications** of hash-to-class group functions?
- More performant class group cryptography libraries are needed!
  - Class group cryptography implemented for ZKP circuits! Pls help!
    - Noir, Cairo, Circom.

"Apply" Slide

- Is there any protocol or application in my firm where I need to sample a random binary quadratic form?
- If we apply a class group verifiable delay function, then likely we need a secure hash-to-class group function!
- Lots of room for efficiency improvement: *class groups are still not ready for most (resource-constrained) applications*.
- The world needs more class group crypto implementation! ***Pls help***!
  - Class group arithmetic implemented for smart contracts, blockchains.
  - Class group arithmetic implemented for zero-knowledge circuits.

RSAC 2025 Conference

Thank you very much!

Q&A

RSAC 2025 Conference

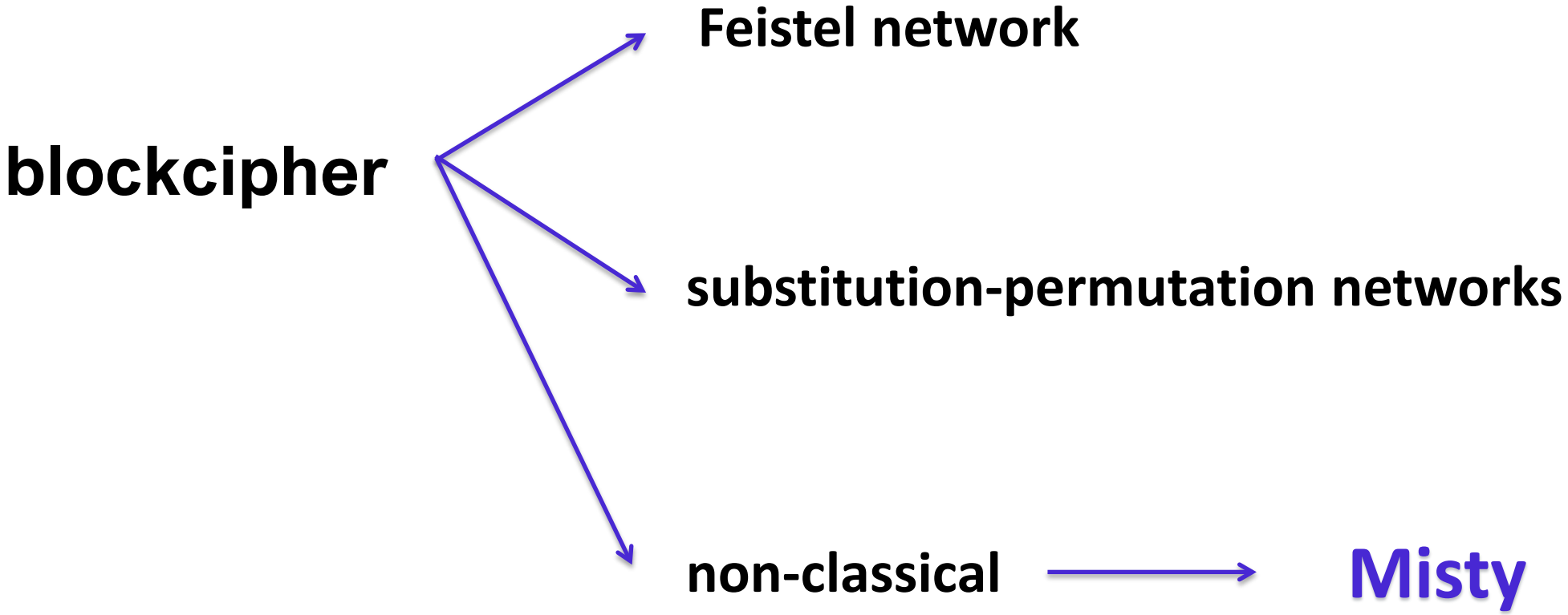
Many Voices.
One Community.

SESSION ID: CRYP-M05B

Sequential Indifferentiability of 7-round Misty Structures

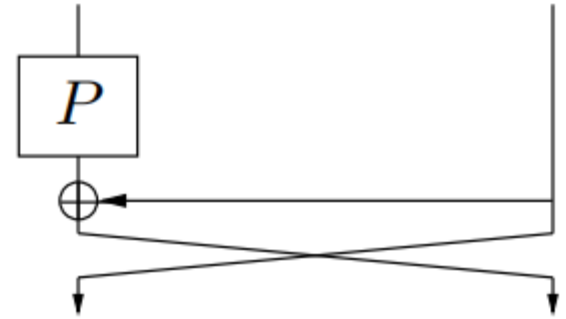
Jiayi Ai

Blockcipher

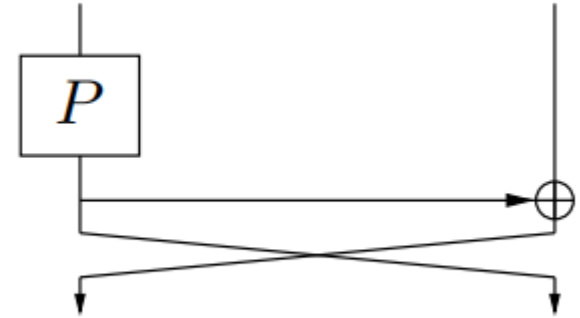


Structure of a Misty work

Misty → **misty-L**, **misty-R**

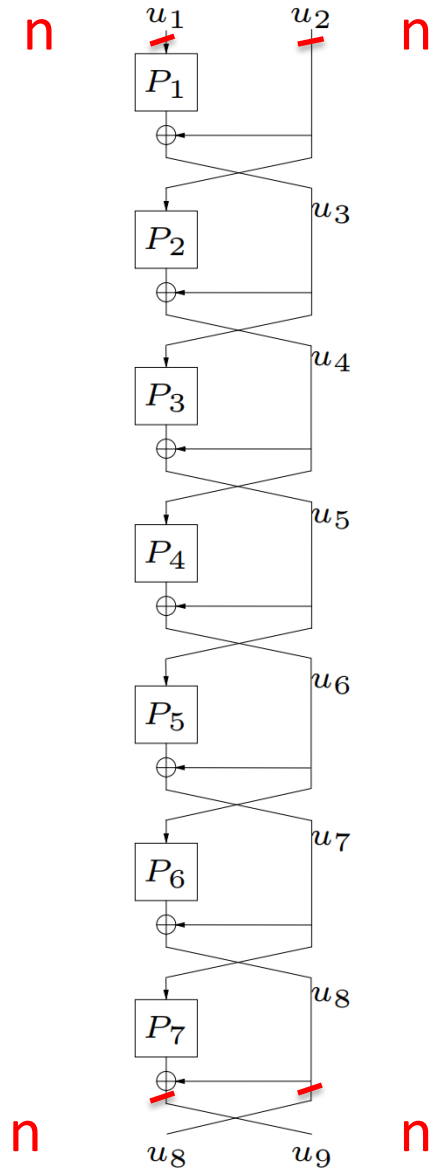


$$\text{Misty-L}^{P}(L, R) := \big(R, P(L) \oplus R\big)$$



$$\text{Misty-R}^{P}(L, R) := \big(P(L) \oplus R, P(L)\big)$$

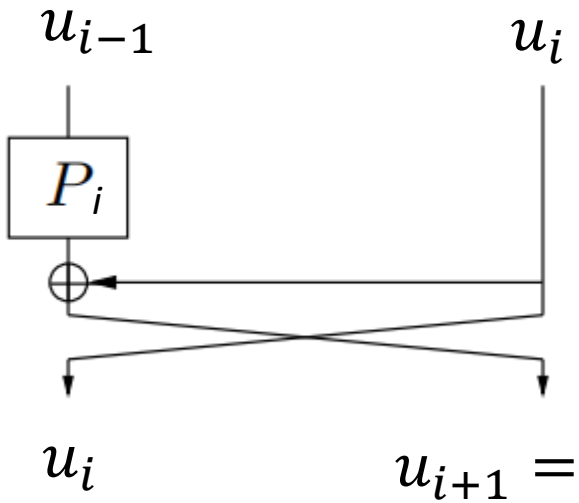
Structure of a Misty work



$$\text{random permutation } P : \{0,1\}^n \rightarrow \{0,1\}^n$$

Each round:

round permutation



$$u_{i+1} = P(u_{i-1}) \oplus u_i$$

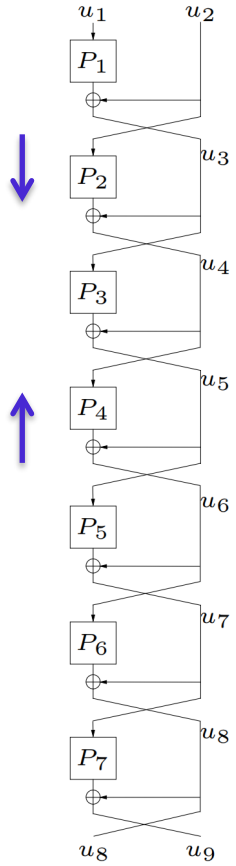
Indistinguishability

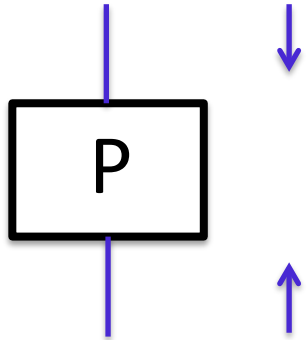
Indifferentiability

D

real world

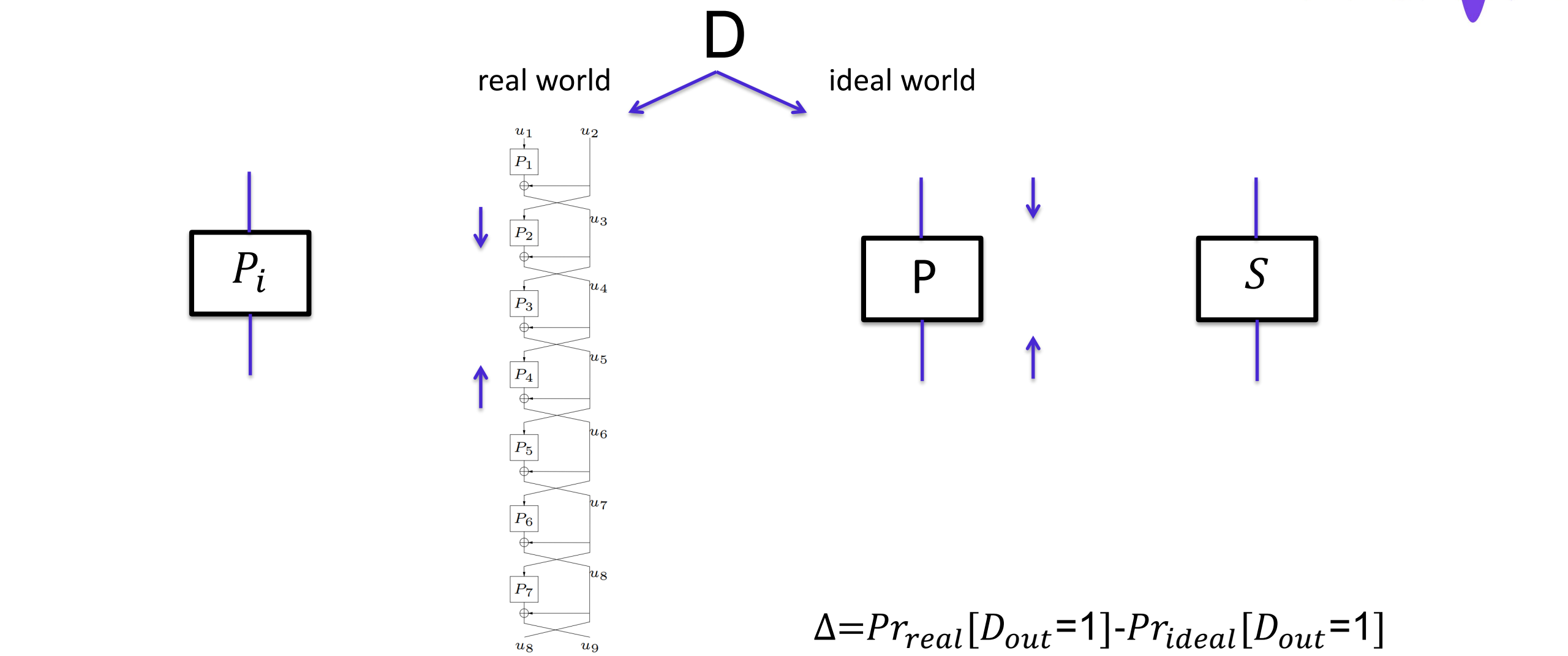
ideal world





$$\Delta = Pr_{real}[D_{out}=1] - Pr_{ideal}[D_{out}=1]$$

Indifferentiability[MRH04]

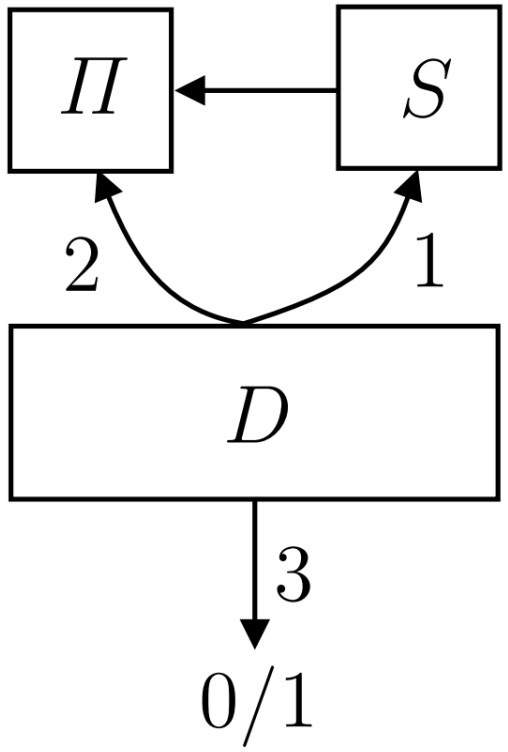


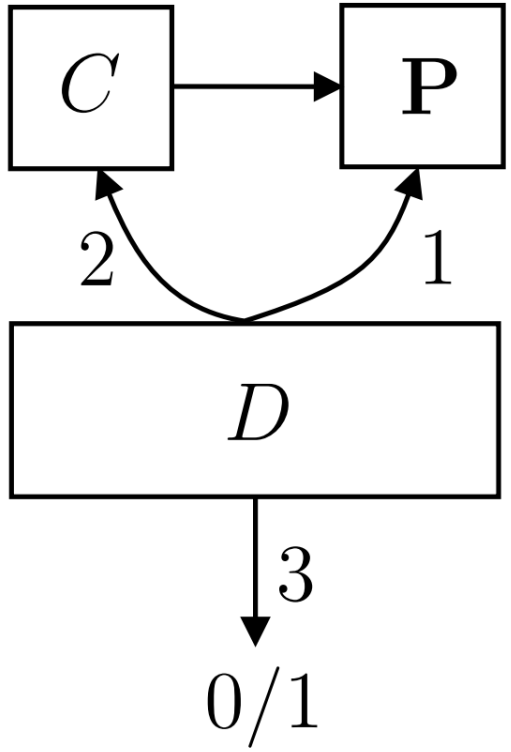
$$\Delta = Pr_{real}[D_{out}=1] - Pr_{ideal}[D_{out}=1]$$

#RSAC
RSAC 2025 Conference

Sequential Indifferentiability

$\{0,1\}^{2n} \to \{0,1\}^{2n}$





$\{0,1\}^{n} \to \{0,1\}^{n}$

Simulator

$\prod_i$ a mapping of entries $(\delta, w) \in \{+, -\} \times \{0,1\}^n$ to values $w' \in \{0,1\}^n$

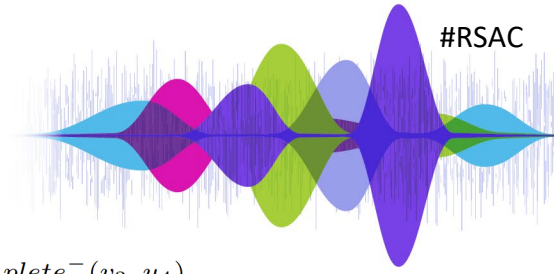
i = 1,...,7

$\text{Query}(i, \delta, w)$ i = 1,...,7

$\delta \in \{+, -\}$ denotes whether the query is direct or inverse

$w \in \{0,1\}^n$ indicates the value being queried

Simulator



1: **Simulator** $S^{\mathbf{\Pi},\mathbf{P}}$
2: **Variables:** tables Π_1,...,Π_7, all initially empty
3: **public procedure** Query(i, δ, w)
4: **if** $(\delta, w) \notin \Pi_i$ **then**
5: $w' := P_i(\delta, w)$
6: $\Pi_i(\delta, w) := w'$
7: $\Pi_i(\overline{\delta}, w') := w$
8: **if** $i = 3$ **then**
9: **if** $\delta = +$ **then**
10: $v_3 = w'$
11: **forall** $u_4 \in \Pi_4^+$ **do**
12: $Complete^+(v_3, u_4)$
13: **else if** $\delta = -$ **then**
14: $v_3 = w$
15: **forall** $u_4 \in \Pi_4^+$ **do**
16: $Complete^-(v_3, u_4)$
17: **else if** $i = 4$ **then**
18: **if** $\delta = +$ **then**
19: $u_4 = w$
20: **forall** $v_3 \in \Pi_3^-$ **do**
21: $Complete^+(v_3, u_4)$
22: **else if** $\delta = -$ **then**
23: $u_4 = w'$
24: **forall** $v_3 \in \Pi_3^-$ **do**
25: $Complete^-(v_3, u_4)$

26: **private procedure** $Complete^+(v_3, u_4)$
27: $v_4 := \Pi_4(+, u_4)$
28: $u_5 := v_3 \oplus u_4$
29: $v_5 := \text{Query}(5, +, u_5)$
30: $u_6 := v_4 \oplus u_5$
31: $u_7 := v_5 \oplus u_6$
32: $u_3 := \Pi_3(-, v_3)$
33: $v_2 := u_3 \oplus u_4$
34: $u_2 := \text{Query}(2, -, v_2)$
35: $v_1 := u_2 \oplus u_3$
36: $u_1 := \text{Query}(1, -, v_1)$
37: $(u_8, u_9) := \mathbf{\Pi}(+, (u_1, u_2))$
38: $v_6 := u_7 \oplus u_8$
39: $v_7 := u_8 \oplus u_9$
40: // Adapt: may overwrite
41: $\Pi_6(+, u_6) := v_6$
42: $\Pi_6(-, v_6) := u_6$
43: $\Pi_7(+, u_7) := v_7$
44: $\Pi_7(-, v_7) := u_7$

45: **private procedure** $Complete^-(v_3, u_4)$
46: $u_3 := \Pi_3(-, v_3)$
47: $v_2 := u_3 \oplus u_4$
48: $v_4 := \Pi_4(+, u_4)$
49: $u_5 := v_3 \oplus u_4$
50: $v_5 := \text{Query}(5, +, u_5)$
51: $u_6 := v_4 \oplus u_5$
52: $v_6 := \text{Query}(6, +, u_6)$
53: $u_7 := v_5 \oplus u_6$
54: $v_7 := \text{Query}(7, +, u_7)$
55: $u_8 := v_6 \oplus u_7$
56: $u_9 := v_7 \oplus u_8$
57: $(u_1, u_2) := \mathbf{\Pi}(-, (u_8, u_9))$
58: $v_1 := u_2 \oplus u_3$
59: // Adapt: may overwrite
60: $\Pi_1(+, u_1) := v_1$
61: $\Pi_1(-, v_1) := u_1$
62: $\Pi_2(+, u_2) := v_2$
63: $\Pi_2(-, v_2) := u_2$

Simulator

Simulator $S^{\mathbf{\Pi},\mathbf{P}}$
Variables: tables Π_1,...,Π_7, all initially empty
public procedure Query(i, δ, w)
  if $(\delta, w) \notin \Pi_i$ **then**
    $w' := P_i(\delta, w)$
    $\Pi_i(\delta, w) := w'$
    $\Pi_i(\bar{\delta}, w') := w$
    if $i = 3$ **then**
      if $\delta = +$ **then**
        $v_3 = w'$
        forall $u_4 \in \Pi_4^+$ **do**
          $Complete^+(v_3, u_4)$
      else if $\delta = -$ **then**
        $v_3 = w$
        forall $u_4 \in \Pi_4^+$ **do**
          $Complete^-(v_3, u_4)$
    else if $i = 4$ **then**
      if $\delta = +$ **then**
        $u_4 = w$
        forall $v_3 \in \Pi_3^-$ **do**
          $Complete^+(v_3, u_4)$
      else if $\delta = -$ **then**
        $u_4 = w'$
        forall $v_3 \in \Pi_3^-$ **do**
          $Complete^-(v_3, u_4)$

private procedure $Complete^+(v_3, u_4)$
  $v_4 := \Pi_4(+, u_4)$
  $u_5 := v_3 \oplus u_4$
  $v_5 := \text{Query}(5, +, u_5)$
  $u_6 := v_4 \oplus u_5$
  $u_7 := v_5 \oplus u_6$
  $u_3 := \Pi_3(-, v_3)$
  $v_2 := u_3 \oplus u_4$
  $u_2 := \text{Query}(2, -, v_2)$
  $v_1 := u_2 \oplus u_3$
  $u_1 := \text{Query}(1, -, v_1)$
  $(u_8, u_9) := \mathbf{\Pi}(+, (u_1, u_2))$
  $v_6 := u_7 \oplus u_8$
  $v_7 := u_8 \oplus u_9$
  // Adapt: may overwrite
  $\Pi_6(+, u_6) := v_6$
  $\Pi_6(-, v_6) := u_6$
  $\Pi_7(+, u_7) := v_7$
  $\Pi_7(-, v_7) := u_7$

private procedure $Complete^-(v_3, u_4)$
  $u_3 := \Pi_3(-, v_3)$
  $v_2 := u_3 \oplus u_4$
  $v_4 := \Pi_4(+, u_4)$
  $u_5 := v_3 \oplus u_4$
  $v_5 := \text{Query}(5, +, u_5)$
  $u_6 := v_4 \oplus u_5$
  $v_6 := \text{Query}(6, +, u_6)$
  $u_7 := v_5 \oplus u_6$
  $v_7 := \text{Query}(7, +, u_7)$
  $u_8 := v_6 \oplus u_7$
  $u_9 := v_7 \oplus u_8$
  $(u_1, u_2) := \mathbf{\Pi}(-, (u_8, u_9))$
  $v_1 := u_2 \oplus u_3$
  // Adapt: may overwrite
  $\Pi_1(+, u_1) := v_1$
  $\Pi_1(-, v_1) := u_1$
  $\Pi_2(+, u_2) := v_2$
  $\Pi_2(-, v_2) := u_2$

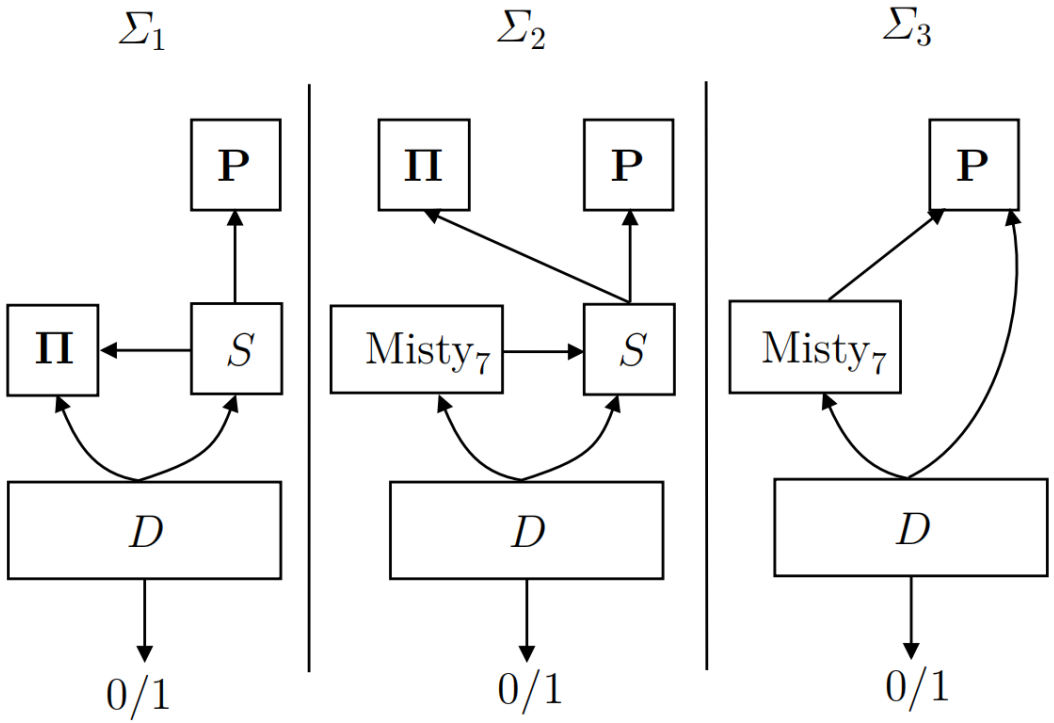
Simulator

Simulator $S^{\mathbf{\Pi},\mathbf{P}}$
Variables: tables Π_1,...,Π_7, all initially empty
public procedure Query(i, δ, w)
  if $(\delta, w) \notin \Pi_i$ **then**
    $w' := P_i(\delta, w)$
    $\Pi_i(\delta, w) := w'$
    $\Pi_i(\bar{\delta}, w') := w$
    if $i = 3$ **then**
      if $\delta = +$ **then**
        $v_3 = w'$
        forall $u_4 \in \Pi_4^+$ **do**
          $Complete^+(v_3, u_4)$
      else if $\delta = -$ **then**
        $v_3 = w$
        forall $u_4 \in \Pi_4^+$ **do**
          $Complete^-(v_3, u_4)$
    else if $i = 4$ **then**
      if $\delta = +$ **then**
        $u_4 = w$
        forall $v_3 \in \Pi_3^-$ **do**
          $Complete^+(v_3, u_4)$
      else if $\delta = -$ **then**
        $u_4 = w'$
        forall $v_3 \in \Pi_3^-$ **do**
          $Complete^-(v_3, u_4)$

private procedure $Complete^+(v_3, u_4)$
  $v_4 := \Pi_4(+, u_4)$
  $u_5 := v_3 \oplus u_4$
  $v_5 := \text{Query}(5, +, u_5)$
  $u_6 := v_4 \oplus u_5$
  $u_7 := v_5 \oplus u_6$
  $u_3 := \Pi_3(-, v_3)$
  $v_2 := u_3 \oplus u_4$
  $u_2 := \text{Query}(2, -, v_2)$
  $v_1 := u_2 \oplus u_3$
  $u_1 := \text{Query}(1, -, v_1)$
  $(u_8, u_9) := \mathbf{\Pi}(+, (u_1, u_2))$
  $v_6 := u_7 \oplus u_8$
  $v_7 := u_8 \oplus u_9$
  // Adapt: may overwrite
  $\Pi_6(+, u_6) := v_6$
  $\Pi_6(-, v_6) := u_6$
  $\Pi_7(+, u_7) := v_7$
  $\Pi_7(-, v_7) := u_7$

private procedure $Complete^-(v_3, u_4)$
  $u_3 := \Pi_3(-, v_3)$
  $v_2 := u_3 \oplus u_4$
  $v_4 := \Pi_4(+, u_4)$
  $u_5 := v_3 \oplus u_4$
  $v_5 := \text{Query}(5, +, u_5)$
  $u_6 := v_4 \oplus u_5$
  $v_6 := \text{Query}(6, +, u_6)$
  $u_7 := v_5 \oplus u_6$
  $v_7 := \text{Query}(7, +, u_7)$
  $u_8 := v_6 \oplus u_7$
  $u_9 := v_7 \oplus u_8$
  $(u_1, u_2) := \mathbf{\Pi}(-, (u_8, u_9))$
  $v_1 := u_2 \oplus u_3$
  // Adapt: may overwrite
  $\Pi_1(+, u_1) := v_1$
  $\Pi_1(-, v_1) := u_1$
  $\Pi_2(+, u_2) := v_2$
  $\Pi_2(-, v_2) := u_2$

Intermediate System



$$\left|\Pr[D^{\Sigma_1(\mathbf{\Pi},S^{\mathbf{\Pi},P})}=1]-\Pr[D^{\Sigma_2(Misty^{S^{\mathbf{\Pi},P}},S^{\mathbf{\Pi},P})}=1]\right|\leq\frac{32q^4}{N}.$$

$$\left|\Pr\left[D^{\Sigma_2(Misty^{S^{\mathbf{\Pi},P}},S^{\mathbf{\Pi},P})}=1\right]-\Pr\left[D^{\Sigma_3(Misty^{P},\boldsymbol{P})}=1\right]\right|\leq\frac{153q^4}{N}.$$

RSAC™ 2025 Conference

Many Voices.
One Community.

Thank You